refactor code snippet to return the following result

console.log(foo.show()); // ["hello world", "hello world"]

var foo = {

title: 'world',

show() {

return ['', ''].map(function(a) { return 'hello ' + this.title });

}

};

Answer:

var foo = {

title: 'world',

show() {

return ['', ''].map(function(a) { return 'hello ' + foo.title });

}

};

console.log(foo.show());

What does the following code print to the console?

function foo() {

var a = 42;

(function() {

a = 27;

console.log(a);

})();

console.log(a);

}

foo();

Answer:

27

27

Given the following code snippet:

let foo = 5;

function addTwo(foo) {

foo = foo + 2;

}

addTwo(foo);

console.log(foo);

Answer: 5

How can you return a new function and assign the this context to an object?

var module = {

x: 42,

getX: function() {

return this.x;

}

}

var var1 = module.getX;

var var2 = var1.bind(module);

console.log(var2());

The following snippet throws an error. Why?

function foo() {

'use strict';

(function () {

var point = {x: 42, y: 27};

with (point) {

console.log('The coordinates are: x: ', x, 'y: ', y);

}

})();

}

foo();

Answer: with statements are not allowed in this mode.

What is wrong with this snippet?

class MegaArray extends Array{

constructor(contents){

this.contents = contents;

}

}

let mArr = new MegaArray(1, 2, 3);

What is the correct syntax for accessing the argument object inside a function?

Answer: Not taught.

----------------------------------------------------------------Excersize------------------------------------------------------------------------------------------------------------------------------------------------------

# Instructions

1. Fix the code so it prints out the alphabet A-Z in the console.

2. Cannot:

- Have \*\*any\*\* global variables at all

- Delete or combine any function declarations

- Create any new functions (except IIFEs -- hint!)

- Rearrange the order of declarations

3. Can/must:

- Declare extra variables (as long as they're not global)

- Modify (in-place) function declaration/initialization

- Add/remove statements/expressions (IIFEs, return, params, etc)

- Make the fewest changes possible

A();

function C() {

console.log("OOPS!");

}

function E(f) {

console.log("E");

f();

var f = F;

}

var A = function() {

console.log("A");

B();

};

var C;

function G() {

console.log("G");

H();

var H = function() {

console.log("H");

I();

};

}

var D = d;

function d() {

console.log("D");

E();

}

function I() {

console.log("I");

J();

J();

}

B = function() {

console.log("B");

C();

};

var F = function() {

console.log("F");

G();

};

var rest = "KLMNOPQRSTUVWXYZ".split("");

for (var i=0; i<rest.length; i++) {

(function(i){

// define the current function

window[rest[i]] = function() {

console.log(rest[i]);

if (i < (rest.length-1)) {

// TODO: call the next function

}

};

})(i);

}

var J = function() {

J = function() {

console.log("J");

K();

};

};

C = function() {

console.log("C");

D();

};